Tutorial: Cross-Validation

Robust Model Evaluation and Selection

ES335 - Machine Learning
IIT Gandhinagar

July 23, 2025

Abstract

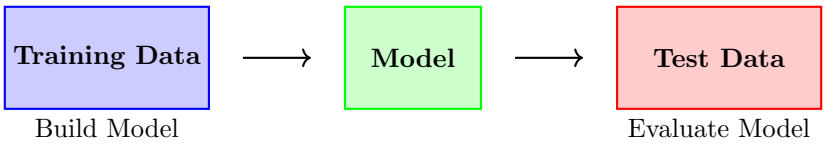
Cross-validation is a fundamental technique for evaluating machine learning models and selecting optimal hyperparameters. This tutorial covers the theory, implementation, and best practices of various cross-validation methods, from basic k-fold to advanced techniques for time series and grouped data. Learn how to avoid common pitfalls and obtain reliable performance estimates for your models.

Contents

1 Introduction: The Model Evaluation Challenge

Imagine you've built a spam email classifier that achieves 99% accuracy on your training data. Is this a good model? Without proper evaluation, you can't tell if your model is genuinely good or just memorizing the training data.

The Core Problem: How do we get reliable estimates of model performance?



Problem: Limited data for both training and testing!

Simple Train/Test Split Limitations:

- Wastes data (typically 20-30% held out)
- Performance depends on particular split
- No systematic hyperparameter tuning
- High variance in performance estimates

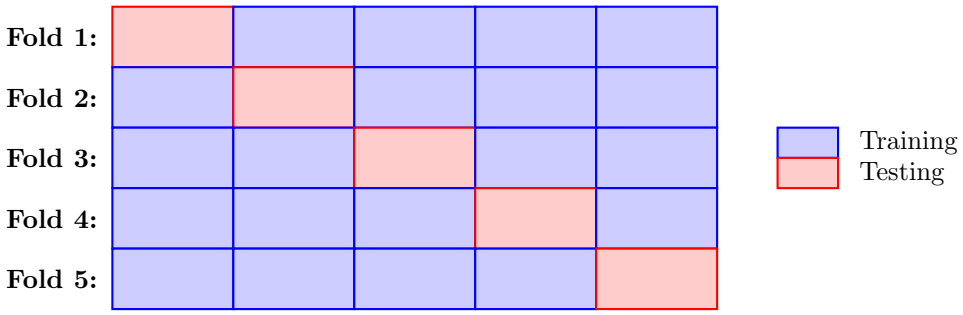
2 K-Fold Cross-Validation

2.1 The Basic Idea

K-fold cross-validation solves the data limitation problem by using each data point for both training and testing, but never simultaneously.

Algorithm:

1. Divide dataset into k equal-sized folds
2. For each fold $i = 1, \ldots, k$:
   - Use fold i as test set
   - Use remaining $k - 1$ folds as training set
   - Train model and evaluate on test fold
3. Average performance across all k folds



2.2 Mathematical Foundation

Let $f^{(-i)}$ be the model trained on all data except fold i, and let D_i be the test fold i.

Cross-validation estimate:

$$\text{CV}_k = \frac{1}{k}\sum_{i=1}^{k} L(f^{(-i)}, D_i)$$

where L is the loss function (e.g., 0-1 loss for classification, MSE for regression).

Standard error:

$$\text{SE} = \sqrt{\frac{\sum_{i=1}^{k}(L_i - \text{CV}_k)^2}{k(k-1)}}$$

Example #1: 5-Fold CV Calculation

Suppose you get these accuracy scores from 5-fold CV: [0.85, 0.82, 0.88, 0.86, 0.84]
Mean accuracy: $\mu = \frac{0.85+0.82+0.88+0.86+0.84}{5} = 0.85$
Standard deviation: $\sigma = \sqrt{\frac{(0.85-0.85)^2+(0.82-0.85)^2+\ldots}{5}} = 0.022$
Report: $85.0\% \pm 2.2\%$ accuracy
This means we're confident the true accuracy is between roughly 82.8% and 87.2%.

3 Cross-Validation Variants

3.1 Leave-One-Out Cross-Validation (LOOCV)

Special case where $k = n$ (number of samples). Each fold contains exactly one sample.

Advantages:

- Maximum use of training data ($n - 1$ samples)
- Deterministic (no randomness in splits)
- Unbiased estimate of generalization error

Disadvantages:

- Computationally expensive (n model fits)
- High variance (each test set is tiny)
- Not suitable for model selection

Mathematical property for linear models:

$$\text{LOOCV} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_{ii}}\right)^2$$

where h_{ii} is the i-th diagonal element of the hat matrix. This allows LOOCV computation with a single model fit!

3.2 Stratified Cross-Validation

For classification problems, regular k-fold might create imbalanced folds. Stratified CV maintains class distribution in each fold.

Example #2: Why Stratification Matters

Dataset: 1000 samples, 90% negative (900), 10% positive (100)
Regular 5-fold: Might create folds with 0-40 positive examples **Stratified 5-fold**: Each fold has exactly 20 positive examples
Without stratification, some folds might have no positive examples, leading to misleading performance estimates!

Algorithm:

1. For each class, divide samples into k groups
2. Combine corresponding groups from each class to form folds
3. Ensures each fold has approximately the same class distribution

3.3 Group Cross-Validation

When samples are not independent (e.g., multiple measurements from same patient), regular CV can leak information.

Examples of grouped data:

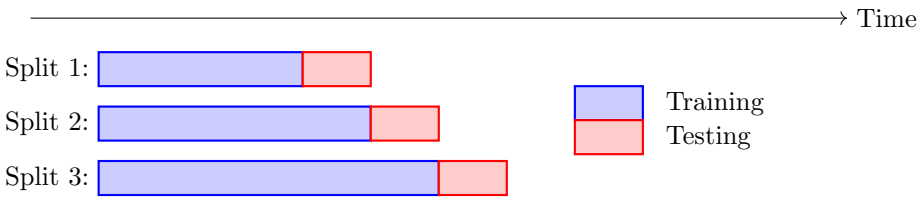
- Medical: Multiple visits per patient
- Finance: Multiple quarters per company
- Time series: Multiple observations per entity
- Images: Multiple crops from same image

Solution: Ensure all samples from the same group are in the same fold.

4 Time Series Cross-Validation

Time series data violates the independence assumption of regular CV. We cannot use future data to predict the past!

4.1 Forward Chaining (Walk-Forward Validation)



Key principles:

- Always train on past data, test on future data
- Training window can be expanding or fixed-size
- Gap between train/test to account for lag

4.2 Blocked Cross-Validation

For time series with seasonal patterns, create blocks that respect temporal structure:

Alternating blocks preserve temporal structure

5 Nested Cross-Validation

When doing both model selection and performance estimation, we need nested CV to avoid optimistic bias.

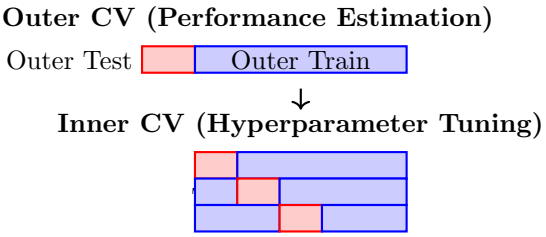
5.1 The Problem with Simple CV

If you use the same CV folds for both hyperparameter tuning and final evaluation, you're indirectly using test data for training!

5.2 Nested CV Solution

Two loops:

- **Outer loop**: Performance estimation (5-10 folds)
- **Inner loop**: Hyperparameter tuning (3-5 folds)



Algorithm:

1. Split data into outer folds
2. For each outer fold:
   - Use outer training set for inner CV
   - Find best hyperparameters via inner CV
   - Train final model with best hyperparameters
   - Evaluate on outer test fold
3. Average outer fold scores for unbiased estimate

6 Common Pitfalls and Best Practices

6.1 Data Leakage

Preprocessing before splitting: WRONG!

```
# Wrong way
data_normalized = normalize(data)
train, test = split(data_normalized)
```

Preprocessing within each fold: RIGHT!

```
# Right way
for train, test in cv_splits:
    train_normalized = normalize(train)
    test_normalized = normalize_with_train_params(test, train)
```

Example #3: The Leakage Problem

Scenario: You're predicting house prices and normalize features using mean/std of entire dataset before CV.
Problem: Test fold statistics influence the normalization of training folds!
Impact: Optimistically biased performance estimates. In extreme cases, can lead to 10-20% overestimation of accuracy.
Solution: Compute normalization parameters only on training folds, apply to test fold.

6.2 Choosing k

Common choices:

- $k = 5$: Good balance of bias and variance
- $k = 10$: More stable, higher computational cost
- $k = n$ (LOOCV): Low bias, high variance, expensive

Bias-variance trade-off:

- **Small k**: Lower variance (more training data), higher bias
- **Large k**: Higher variance (less training data), lower bias

6.3 Statistical Significance

Comparing models: Use paired t-test on CV fold scores

$$t = \frac{\bar{d}}{\text{SE}(\bar{d})} = \frac{\bar{d}}{s_d/\sqrt{k}}$$

where d_i is the difference in performance between two models on fold i.
Confidence intervals:

$$\mu \pm t_{\alpha/2,k-1} \times \frac{s}{\sqrt{k}}$$

7 Advanced Topics

7.1 Repeated Cross-Validation

Repeat CV multiple times with different random splits to get more stable estimates:

$$\text{Repeated CV} = \frac{1}{r}\sum_{j=1}^{r} \text{CV}_j$$

where r is the number of repetitions. Reduces variance but increases computational cost.

7.2 Bootstrap Methods

Alternative to CV using resampling with replacement:

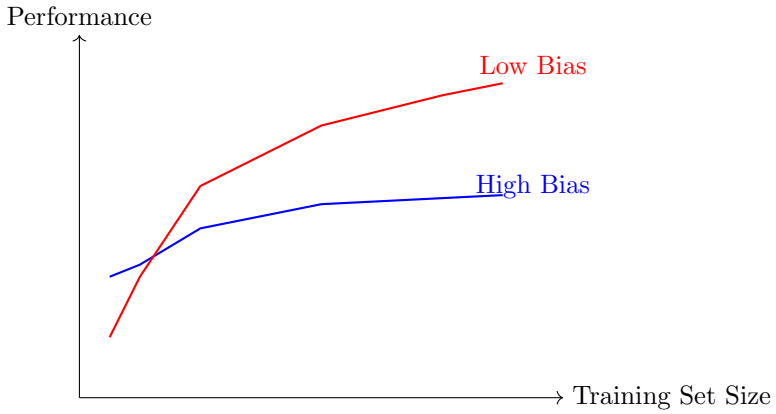
Bootstrap .632 estimator:

$$\text{Err}_{.632} = 0.368 \times \text{Err}_{\text{train}} + 0.632 \times \text{Err}_{\text{boot}}$$

Addresses optimistic bias of bootstrap by combining training error and out-of-bag error.

7.3 Learning Curves

Use CV to understand how performance scales with training set size:



More data helps low-bias models more

8 Implementation Guidelines

8.1 Computational Considerations

Time complexity: $O(k \times T)$ where T is model training time **Space complexity**: Usually $O(1)$ if models aren't stored

Optimization strategies:

- Parallel fold evaluation
- Early stopping for clearly bad hyperparameters
- Warm starts for iterative algorithms
- Cached preprocessing

8.2 Practical Workflow

1. **Choose CV strategy** based on data characteristics
2. **Set up preprocessing pipeline** that fits within folds
3. **Define hyperparameter grid** for tuning
4. **Run nested CV** if doing model selection
5. **Analyze results** with confidence intervals
6. **Validate on final holdout** set

9 Practice Problems

9.1 Basic Problems

Problem #1: CV Score Calculation

You perform 5-fold CV on a binary classification problem and get these confusion matrices:
Fold 1: TP=18, FN=2, FP=3, TN=17
Fold 2: TP=16, FN=4, FP=2, TN=18
Fold 3: TP=19, FN=1, FP=4, TN=16
Fold 4: TP=17, FN=3, FP=1, TN=19
Fold 5: TP=15, FN=5, FP=2, TN=18
Calculate the mean accuracy and 95% confidence interval.
Solution: Fold accuracies: [0.875, 0.85, 0.875, 0.925, 0.825] Mean = 0.87, Std = 0.0374 95% CI: $0.87 \pm 1.96 \times 0.0374 = [0.797, 0.943]$

Problem #2: Choosing the Right CV

For each scenario, choose the most appropriate CV method:
a) Predicting daily stock prices using past 5 years of data
b) Classifying medical images with 10,000 samples from 100 patients
c) Binary classification with 95% negative, 5% positive examples
d) Small dataset with 50 samples for hyperparameter tuning
Solutions: a) Time series CV / Forward chaining b) Group CV (group by patient) c) Stratified CV d) LOOCV or repeated CV

9.2 Advanced Problems

Problem #3: Nested CV Analysis

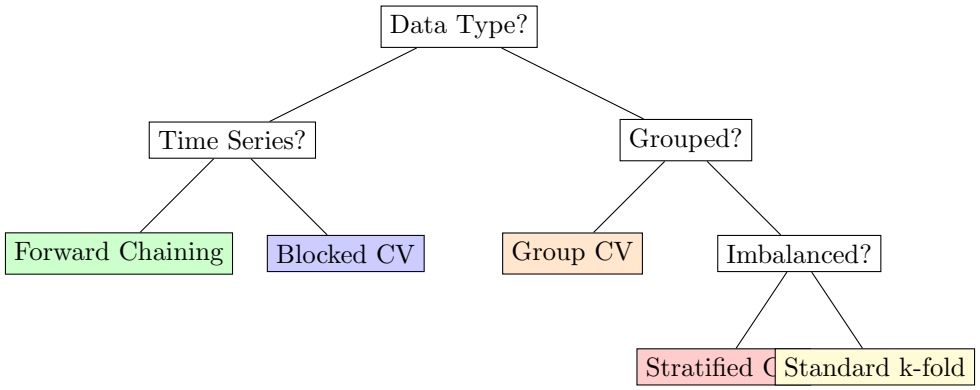
You have 1000 samples and want to: 1. Compare 3 algorithms 2. Tune hyperparameters for each 3. Get unbiased performance estimate
Design a nested CV strategy with computational budget considerations.
Solution: - Outer CV: 5-fold for final comparison - Inner CV: 3-fold for hyperparameter tuning - Total model fits: $5 \times 3 \times 3 \times H$ where H is hyperparameter combinations - Use parallel processing and early stopping to manage computational cost - Report performance as mean $\pm$ std across outer folds

10 Summary and Guidelines

10.1 Key Takeaways

1. **Choose CV method based on data structure**: Regular k-fold for i.i.d. data, stratified for imbalanced classes, time series CV for temporal data, group CV for clustered data
2. **Avoid data leakage**: Always preprocess within CV folds, never on the entire dataset
3. **Use nested CV for model selection**: Prevents optimistic bias when tuning hyperparameters
4. **Report confidence intervals**: Mean performance alone is insufficient; include variability estimates
5. **Validate final model**: Use a separate holdout set for final validation after all CV-based decisions

10.2 Decision Tree for CV Method Selection



10.3 Best Practices Checklist

- ☐ Choose appropriate CV method for your data structure
- ☐ Use stratification for classification problems
- ☐ Implement preprocessing within CV folds
- ☐ Use nested CV for hyperparameter tuning
- ☐ Report mean ± standard deviation/confidence interval
- ☐ Check for statistical significance when comparing models
- ☐ Reserve final holdout set for ultimate validation
- ☐ Consider computational budget when choosing k

11 Further Reading

- **Comprehensive Review**: Arlot & Celisse "A survey of cross-validation procedures for model selection" (2010)
- **Time Series CV**: Hyndman & Athanasopoulos "Forecasting: Principles and Practice"
- **Statistical Analysis**: Dietterich "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms" (1998)
- **Practical Guide**: Kuhn & Johnson "Applied Predictive Modeling"
- **Modern Implementation**: scikit-learn cross-validation documentation